LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

UCRL-TR-220342-REV-1

ASC Predictive Science Academic Alliance Program Verification and Validation Whitepaper

R. Klein, S. Doebling, F. Graziani, M. Pilch, T. Trucano

April 19, 2006

Disclaimer

Auspices Statement

This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

Updated October 14, 2003

UCRL-TR-220342-REV-1

March 29, 2006

ASC Predictive Science Academic Alliance Program Verification and Validation Whitepaper

Richard Klein[1], Scott Doebling[2], Frank Graziani[1], Marty Pilch[3], Tim Trucano[3]

1. Lawrence Livermore National Laboratories
2. Los Alamos National Laboratory
3. Sandia National Laboratory

1. Introduction

The purpose of this whitepaper is to provide a framework for understanding the role that verification and validation (V&V) are expected to play in successful ASC Predictive Science Academic Alliance (PSAA) Centers and projects. V&V have been emphasized in the recent specification of the PSAA (NNSA, 2006):

- "The resulting simulation models lend themselves to practical verification and validation methodologies and strategies that should include the integrated use of experimental and/or observational data as a key part of model and sub-model validation, as well as demonstrations of numerical convergence and accuracy for code verification."
- "…verification, validation and prediction methodologies and results must be much more strongly emphasized as research topics and demonstrated via the proposed simulations."
- "It is mandatory that proposals address the following two topics:
    1. Predictability in science & engineering.
    2. Verification & validation strategies for large-scale simulations, including quantification of uncertainty and numerical convergence."

We especially call attention to the explicit coupling of computational predictability and V&V in the third bullet above. In this whitepaper we emphasize this coupling, and provide concentrated guidance for addressing item 2.

The whitepaper has two main components. First, we provide a brief and high-level tutorial on V&V that emphasizes critical elements of the program. Second, we state a set of V&V-related requirements that successful PSAA proposals must address.

2. Overview of V&V

2.1 Definitions

It is important to consider three statements in the latest Advanced Simulation and Computing (ASC) Program Plan of the United States Department of Energy's (DOE) National Nuclear Security Administration (United States Department Of Energy, 2003).

- The Mission of the ASC program is to "Provide leading edge, high-end simulation capabilities needed to meet weapons assessment and certification requirements."
- The Vision of the ASC program is to "Predict, with confidence, the behavior of nuclear weapons, through comprehensive, science-based simulations."
- The Strategic Goal of the ASC program is to provide "Predictive simulations and modeling tools, supported by necessary computing resources, to sustain long-term stewardship of the stockpile."

Additionally, it is stated in this DOE program plan that "… [V&V] will provide high confidence in computational accuracy by systematically measuring, documenting, and demonstrating the predictive capability of codes…" These quotes define high expectations from V&V within the ASC program and projects it funds.

The definitions of V&V stated in this same document are:

> **Verification**: Verification is the process of confirming that a computer code correctly implements the algorithms that were intended.
>
> **Validation**: Validation is the process of confirming that the predictions of a code adequately represent measured physical phenomena.

These definitions have a relatively straightforward operational interpretations (see Roache, 1998 for further discussion of this point) that ends up guiding all the key elements of V&V for ASC.

> **Verification**: The process of confirming that the equations are numerically solved accurately.
>
> **Validation**: The process of confirming that the equations are (physically) accurate.

To avoid unbounded philosophical problems associated with inductive inference in the use of such computational models, we claim that this interpretation of validation is targeted on specific applications of the models. (ASC, and the PSAA, emphasize *predictive* applications.) This tends to narrow the focus of this interpretation of verification as well. However, since verification is at its core a mathematical problem, there must be a greater element of generality associated with it than with validation. The key insight, however, is that results of V&V are not generally true for the code independent of the targeted application.

General references that may be of aid to the readers of this document include Roache (1998), Oberkampf and Trucano (2002), Oberkampf, Trucano, and Hirsch (2004). Another useful document is the V&V guide developed by AIAA (1998).

With this introductory background, we now turn to specific discussion of elements of V&V.

2.2 Verification

2.2.1 Code verification

Code verification is the most general component of V&V. It answers, or seeks to answer, three specific questions: (1) Are the equations represented by a code *mathematically* (not physically) correct? (2) Are the algorithms that provide the numerical solution of the mathematical equations themselves mathematically correct? (3) Is the software implementation of these algorithms correct (that is, free of faults)? Resolution of these questions, if possible, is quite general and in principle does not depend on specific applications of the code. However, code verification does not directly address the problem of deciding how numerically accurate a given calculation is, that is calculation verification (or calculation numerical error assessment), but it provides the necessary foundation for believing accuracy assessments. The specific accuracy achievable in given calculations is very much a function of the particular application.

Often the equations to be solved are accepted in their given mathematical form and attention immediately turns to numerical solution algorithms. Where the mathematical form of the equations to be solved is open to questions, or subject to manipulation prior to the development of solution algorithms, these manipulations are part of the code verification problem. One typically addresses the mathematical correctness of the equations through mathematical proof.

Similarly, in principle algorithms that are constructed for the numerical solution of the underlying equations are also verified via rigorous mathematical proof. However, it is well-known that complete mathematical rigor often cannot be applied to many important computational physics algorithms. The code verification problem becomes very complex because a great deal of assessment of the "correctness" of numerical algorithms is empirical, that is, it depends on observed performance of the algorithms on a wide variety of problems. In other words, algorithm correctness assessment is very much a problem of testing.

Software implementations that define the code are verified through the accumulation of evidence of error-free functioning, primarily through software development methods used, and through testing. SQE (Software Quality Engineering) is the general body of practice that speaks to developing software using techniques that minimize the creation of software faults. It does not guarantee the absence of such bugs, no matter how rigorously it may be pursued. Testing distinct from formal SQE methodologies remains an important assessment principle for verification of software implementations in computational science and engineering. And, as we noted above, in practice testing is critically related

to verification of algorithms. See Oberkampf and Trucano (2002), Oberkampf, Trucano, and Hirsch (2004) for more discussion of these issues.

We will not discuss SQE in this paper. We will address a few remarks to testing. Finding or developing new test problems that fully address the complex couplings in the kinds of computational science projects funded by ASC is very difficult (for example, consider the paucity of test problems for coupled radiation-hydrodynamics, multi-dimensional hydrodynamics with wave interactions, realistic equations of state, other energy transport mechanisms, such as neutron transport, or plasma thermal conduction).The typical approach to testing is to test isolated components of the physics to the degree possible, and use experience (judgment) sanity checks (such as energy and momentum conservation and preservation of symmetries), and robustness evaluation (does the code even run when all the physics is turned on?) to test coupled physics.

It is useful to think about verification testing and test problems in three dimensions. These dimensions measure independent categories of knowledge that should always be associated with testing. The first of these dimensions is the ***structure*** of the chosen test problems, that is, the logical principles underlying them. This dimension addresses the important question of why given test problems are chosen, and how they are organized. This is important in mapping the kind of testing that is typically performed in computational science to more formal characteristics of software testing seen in software engineering, for example, concerns over how a test suite "covers" the targeted code.

The second dimension is the specific ***construction*** of the test suite, or the specific means chosen by the code team for populating the test problem suite. Finding or developing new test problems that fully address the complexities of multiphysics codes is a tremendous challenge. This is an important point of collaboration between the PSAA projects and ASC laboratories. It is also important to provide detailed, unambiguous specifications of all test problems used for code verification. This allows accurate replication of the tests by the project as well as the broader computational community.

The third dimension of importance for verification test problems is that of ***assessment***, specifically the criteria that are applied for deciding whether or not the code has passed or failed a given test problem. Verification test problems are intended to be strong tests of the code. Therefore, assessment must be objective and rigorous, and well-documented.

Specification of verification tests in these three dimensions – why tests are chosen, how they are formulated, and how they are assessed - is an important part of PSAA projects.

An option for verification testing that is sometimes used is so-called code comparisons, in which one code is (ideally) chosen as a referent, and compared with a different code on one or more calculations. With code comparison one wants to test the code that has improved physics against the legacy code that has older physics by running the code with improved physics at the same level of approximation in the legacy code and comparing results. Trucano, Pilch, and Oberkampf (2003) analyze both logical difficulties and execution problems in the normal use of code comparisons and argues that this is, at best, a poor contribution to V&V. Critical to the success of this approach is to have a firm basis of evidence for the use of an alternative code as a referent, that is a benchmark, and

a firm basis for assessing the comparison. Both are often lacking in many code comparison studies that we have observed.

2.2.2 Calculation (solution) verification

Solution verification is quantification of the numerical error in a presented calculation. This answers a direct question: What is the error in a given calculation? Unfortunately, this is all but impossible to perform completely and rigorously for complex calculations. However, it can be partially and practically addressed by explicit discretization robustness and convergence studies, formal error estimation procedures, inference from test problem suites, and – possibly with some danger – inference from previous experience (i.e. judgment). Past experience can count for much *if properly understood and presented*.

Code verification can be completely achieved, and calculations can still be inaccurate, due to poor discretizations (lack of converged calculations). More generally, verification of correct functioning of algorithms cannot be partitioned as cleanly as we would like. It may be impossible to determine that algorithms are failing only on available test problems; the failures may appear only on large-scale problems for which there is no referent solution. In such a case, the only warning that the solution is incorrect will be heuristic or empirically based assessment of the quality of specific calculations, which is what calculation verification is about.

In validation, explicit solution verification must be performed. It targets the numerical errors present in any comparison of a calculation with experimental data. The fundamental question that must be recognized, if not completely answered, is "Does the numerical error fatally corrupt the comparison with experimental data?" In the absence of acknowledgment of this problem, comparison with experimental data is irrelevant.

A common fallacy is to effectively ignore the problem, observe good agreement with experimental data by means of the chosen comparison, and then conclude that *numerical* accuracy is good. *Mathematical accuracy is not measured by comparison with experimental data.* One simple counter example is the presence of mutually canceling bugs in a code that happen to lead to fortuitous agreement with selected data. Another example is to observe agreement with experimental data at one mesh or discretization resolution, and then see the agreement *worsen* as the mesh is resolved. If mesh refinement studies are not performed this problem will never be observed.

Calculation verification, in the absence of completely rigorous mathematics applicable to the full scope of the PDEs being solved, is essentially empirical. The key procedures that offer promise are: (1) a posteriori error estimation; (2) convergence studies; (3) numerical error models; (4) uncertainty quantification methods treating the numerical error as an epistemic (lack-of-knowledge) uncertainty.

2.3 Validation

2.3.1 Experimental Validation

Validation is fundamentally an experimental challenge. The equations that are solved in ASC codes are determined to be physically accurate (for a given application) through confrontation with experimental data having quality suitable for achieving the goals of validation. ASC generally refers to the process of experimental validation as the core challenge of the V&V program. The requirements and constraints on experimental validation for ASC have been documented in various internal documents, as well as partially in the literature cited in Section 2.1. For example, the experimental validation process that the ASC V&V program deploys at SNL is defined and carefully discussed in Trucano, Pilch, and Oberkampf (2002).

Because of limited resources, it is important to prioritize validation tasks. The logical desire to achieve complete validation of a complex code for a predictive complex multi-physics application must be balanced against these constrained resources. Some kind of planning is required to achieve this, and is almost certainly required to properly collaborate with a program of validation experiments. We implicitly assume that this kind of planning has been achieved, and the results summarized as what needs to be done, what will be done given the resources, and what will likely remain incomplete. The prioritization must proceed from a critical analysis of the key physics required in the driving application(s).

Validation centers on validation calculations, which must be verified to the degree possible as discussed in Section 2.2. Key elements (mainly necessary, but not claimed to be sufficient) of experimental validation, are inevitably:

1. Precise specification of the needed validation tests to optimize the alignment of validation calculations with executed experiments. This requires sophisticated two-way communication between those who execute validation experiments and those who perform validation calculations. This is one reason that ASC places prominence on dedicated validation experiments over, say, experimental data found in the literature. Ambivalence in the experimental data – for example, making the claim "We need to do calculations to understand the experimental data" – implies the experiment is not a validation experiment. More emphatically, validation is weakened when experimental data are not validation quality. The expectation is that the experiments themselves have been subjected to verification and validation to provide the highest quality data. That is, **experiment verification** confirms that the experiment was executed correctly; **experiment validation** confirms that the correct experiment was executed.

2. Performance of calculation verification for all validation calculations as indicated in Section 2.2.

3. Quantification of measurement/computational prediction comparisons, including quantified uncertainty. This *requires* (a) experimental error bars that encompass experimental uncertainty and (b) calculation error bars that encompass calculation uncertainty determined by a program of simulation uncertainty quantification (cf. section 2.4).

4. Assessing the results of validation in terms of computational credibility for the intended application. In the case of ASC, and because of the emphasis on predictive capability in the PSAA, this assessment must address what the completed validation work has told us about *predictive capability*.

2.3.2 Validation Planning Logic

The validation tasks that are necessary and sufficient for any particular code application should emerge naturally as an artifact of careful V&V planning. The logical order of important relationships expressed in this planning is crudely summarized as:

Rigorous scrutiny of physical phenomena required for intended application

→ Validation Priorities

→ Needed validation experiments

→ Needed validation calculations

→ Revise validation plan

An example of guidance for planning validation is Pilch, et. al, (2000). Understanding the simulation requirements and their relationship to the important physical phenomena required for the intended application must be transformed into specific prioritized validation requirements. The results of this are "Needed validation experiments" and "Needed validation calculations." All subsequent validation work responds to these needs. Of course, we also recognize the fact that as validation proceeds, the original assessment of required physical phenomena and what has the highest priority for validation can change as a consequence of accumulated validation information. Thus, we acknowledge the likely need for revision of the defined validation tasks as a result of executed work.

Validation experiments, that is, the high quality data they provide, are *required* to execute this logic. Validation planning should reveal that some of the required physical phenomena are perceived to be adequate for the application, or are viewed for other reasons as having low priority, and thus are not expected to be addressed in the planned experimental validation. The main reasons that are acceptable for this are either because these phenomena are known to be "Previously validated" (meaning that this work was performed to the characteristics specified in this paper) or because they are identified as being mainly irrelevant to the intended application. This is a judgment that, of course, can quickly change as the project progresses.

A third possibility is troublesome – that an identified phenomenon is important, but that there are not resources to attack it (i.e. we can't afford to perform a validation experiment), or that there are scientific reasons that validation cannot be pursued (i.e. an experiment cannot be performed). This is a ***gap in the validation logical structure***. This kind of gap has implications for the ultimate application of the code, and constrains how rigorous our validation assessments can be. It is essential to communicate this kind of gap.

One option that is always considered in a resource constrained world is to use existing experimental data. Considerable caution should be exercised when using non-validation-dedicated experimental data for high-quality validation (Trucano, Pilch, and Oberkampf, 2002). At the very least, validation planning should provide a basis for establishing data requirements, so that existing data, as well as dedicated data gathered by directed validation activities, can be assessed as to its quality for validation. The absence of appropriate data implies, *of course*, "No Validation."

It is essential that required (that is high priority) validation tasks emerging from the planning process be carefully documented. We place significance on the opportunities presented by dedicated validation experiments for the engagement of modeling in the predictive design and analysis of validation experiments, not just passive post-experiment comparisons. This level of engagement with dedicated experiments can only be accomplished from a sound planning basis that is documented.

The goal of good validation planning is retrospective as well as forward looking. Not only does a rigorous and prioritized plan provide a mechanism to generate validation tasks. It also provides a means of connecting accomplished validation work to the overall logical fabric of a project, especially critical for important predictive applications.

2.3.3 Validation Calculations

Validation calculations are calculations that are compared with validation quality experimental data for the purpose of inferring physical accuracy of the associated calculations. The ASC program is particularly concerned with "predictive" modeling, so the intent of validation under the ASC program is to assess "predictive capability" of a code for a stated application. ASC funded V&V projects address the credibility of predictions made by ASC codes for applications of interest to NNSA, specifically in support of Defense Programs activities. This is the purpose of stressing V&V in the PSAA.

Validation calculations have the specific purpose of enabling an assessment of the physical quality/physical accuracy/predictive capability of the code for the application represented by the chosen validation data. The ***experimental data*** that validation calculations are compared with must have specific characteristics in order to be effective in enabling validation. These characteristics include quantified experimental uncertainty, reproducibility and robustness of experimental data, and as directly comparable with calculations as possible. This latter point means that we don't have the situation where "apples" are measured, "oranges" are directly calculated, and the two must somehow be

compared. A detailed discussion of needed validation experimental data characteristics is presented in Trucano, Pilch and Oberkampf (2002). As we previously mentioned, not all experimental data can be considered to be useful for validation. Furthermore, not all comparisons with appropriate experimental data can be considered to be validation in the precise sense that is defined by the above paper.

Comparisons of calculations and experiments for the purpose of validation require a quantitative understanding of the presented comparison, which is often in the form of plots, but could also be detailed tabular comparisons or other quantitative representations of the comparison. In particular, this means that the *uncertainty in the experimental data* and the *uncertainty in the presented calculation(s)* must be acknowledged and accounted for in the details of the comparison. These factors influence the conclusions that can be drawn from the validation comparisons. *These necessary factors in validation comparisons significantly raise the bar on the logic and discourse associated with comparing calculations with experimental data.*

The "science" of performing experimental-computational comparisons in "computational science" remains immature (Oberkampf and Trucano, 2002; Oberkampf, Trucano, and Hirsch, 2003). An important goal of the V&V program is to advocate work that improves the "science" of these comparisons, and thus strengthens the conclusions that can be drawn from them. The issue is, in fact, larger than ASC and should be perceived as critical to the goals of computational science as a truly rigorous discipline.

2.3.4 Experimental Error Bars

Experimental "***Error bars***" is a euphemism for "experimental uncertainty quantification." This is another problem that is unlikely to be completely and rigorously solved for complex experiments. The components of error bars are experimental bias and variability, and various factors in real experiments enter into these components.

The presentation of experimental error bars can literally be error bars on plots of experimental data. It can also be a precise discussion of what is known about that error bar. A plot that contains a calculation compared with an experiment in which no experimental "error bar" is presented or discussed invites one of two interpretations: (1) either the "error bar" is the size of the plot symbol (width of the experimental curve); or (2) the "error bar" is the size of the plot. We favor the latter interpretation in the absence of needed information. An error bar of this magnitude implies that the calculated comparison is then meaningless. This emphasizes our point – validation calculations *without* reported experimental error bars are essentially meaningless (for validation anyway).

To perform validation, some approximation to experimental "error bars" must be accomplished and presented to serve as a starting point for inference about the experimental-computational comparisons. Gross contributions to experimental uncertainty are diagnostic fidelity, experimental variability, and experimental bias. The *minimum necessary information* that should reasonably be expected for any candidate for validation data is diagnostic resolution characterization. Repeat experiments help

quantify experimental variability. Experimental bias and substantive lack-of-knowledge uncertainty (which is related to the broad question raised earlier of whether the experiment is the "correct experiment") can seemingly be addressed only through the mechanism of multiple facilities.

The more we expect to rigorously infer from a validation comparison, the more we need to understand about experimental error bars as quantifications of experimental uncertainty. For example (and this is a trivial example) – is an experimental error bar a central tendency of an underlying Gaussian distribution, a statistical confidence interval, a representation of a uniform distribution, a possibility interval, or something else again?

A recent reference on this topic is Rabinovich (2005).

2.4 Uncertainty Quantification

The quantification of uncertainty (UQ) in large scale simulations is playing an increasingly important role in the process of code verification and validation. If a simulation is to be quantitatively validated against the results from an experiment, it is crucial to understand the expected uncertainty in the output metrics of the calculation and also have a quantitative determination of the error bars associated with the output metrics from the experiment. In practice, it becomes possible to assess the true accuracy of a simulation when the experimental uncertainty is less than the predicated uncertainty of the simulation. Error estimates of uncertainty for the experiment usually require that an ensemble of experiments with controlled parameters be performed and known systematic errors are understood.

The quantification of uncertainty in large scale simulations becomes particularly important when the simulation is used as a predictive tool in describing phenomena in a regime that is outside of the bounds of previous experimental tests or known observations. Without experiments to check against code predictions in such regimes, it becomes essential to quantitatively evaluate the expected uncertainty in code output. This task of UQ is complex in its undertaking for any simulation code that has non-linearly coupled multi-physics algorithms as a representation of the underlying partial differential equations.

In a complex multi-physics simulation code, many aspects of the physics may have a parametric representation or a choice of physics models each with their own degree of approximation. The range or bounds of parametric settings in physical models and the choice of physics models represents a span of uncertainty in the simulation. Typically, simulation codes are used with a particular choice of input physics models and perhaps a typical choice of parametric settings without any exploration of the full uncertainty in the simulation outcome. Occasionally, a few different models are run in a few large scale simulations to uncover an estimation of the range or dispersion of output results and this gives some measure of the uncertainty, but it is usually woefully inadequate for determination of the full uncertainty in the simulation. The problem of determination of uncertainty quantification is complex and is a topic for current research. Every potential

center that has a strong V&V component should have a part of that V&V component devoted to determination of UQ of it simulations.

To start, one must first identify the known sources of uncertainty in the simulation. This may involve uncertainties associated with approximate models for the underlying physics, (eg. Sub-grid turbulence models; flux-limited diffusion for radiation transport, flame front propagation models etc.); approximations in the numerical algorithms used; uncertainties associated with the settings of parameters that are used in physical models; settings that individual algorithms may have to work in a stable fashion; uncertainties associated with various levels of opacity tables, equation of state tables, and of course uncertainties associated with performing the simulation at a given spatial resolution when this resolution is not converged. Considering that a multi-physics code embodies many components of coupled physics, the list of possible sources of simulation uncertainty can be quite large. Moreover, the uncertainties associated with these sources do not combine linearly, but may take on combinatorics of all possible settings. Furthermore, uncertainties associated with various physics models within the code may cancel giving compensating effects. As a simple example, let us consider a single physical model used in a code that is parametrically represented by N parameters each of which may have any of M settings within well prescribed bounds for each of the N parameters. If the settings of each of the N parameters can take on M settings with some probability density function for the likelihood of any of the settings, then the total number of possible settings is M^N. To ascertain the full uncertainty in the outcome of the simulation or the dispersion of possible results, one would have to perform M^N possible simulations. If the physical model in question had 5 parameters each of which could take on 10 possible values, in principle 100,000 simulations would have to be performed to get the full range of uncertainty on the output metric from the code. If the number of parameters representing the physical model were to increase to 6, the number of simulations would expand to 1,000,000. This is clearly computationally prohibitive for 2-D simulations and not currently possible for 3-D on any existing terascale platform . In a realistic multi-physics, multi-dimensional code, the number of parameters whose values may be bounded may be large and the problem of examining the full possible uncertainty resultant from all possible non-linear interactions among the uncertain components becomes exponentially complex. The problem of Uncertainty Quantification becomes one of reducing the computation of the full uncertainty space by a huge factor to become computationally tractable (Saltelli, Chan, and Scott, 2000).

The first step in an approach to Uncertainty Quantification is to identify all avenues of certainty for the simulation code. Once this is established, some approach to the development of a sensitivity analysis must be developed to determine which components of uncertainty (algorithmic approximation, parameters, etc.) are the dominant drivers of the output metrics. This is likely to be an iterative process that cannot be determined a priori. In order to perform a sensitivity study to filter out those components of uncertainty that may not dominate the total output uncertainty, one required to know the acceptable bounds of any set of parameters that represent a physical model. The determination of physically reasonable bounds may require a considerable research effort and the quantification of such bounds may be possible with knowledge gained from experiments, analytic analysis and scientific judgment. Given a first estimate of the

sensitive drivers of the code's response to parametric and physical model variation, the problem can now be viewed as navigating the uncertainty of these dominant drivers in an N dimensional space where each dimension is representative of a parameter, physical model, degree of approximation etc. to the underlying code physics. It becomes essential to sample the full N-dimensional space with a set of simulations that are representative of all dimensions of uncertainty within the bounds of those dimensions. Thus the problem of uncertainty quantification becomes one in which all identifiable uncertainties and their interaction with one another are run through the simulation code to give a predictable total output uncertainty in the code's response to variation over acceptable bounds of all the components. The uncertainty in code response to uncertainties in all the key components of the code can be expressed as the total uncertainty in the main metrics of code output that are objectives of the simulation.

3. Critical Issues

3.1 Verification

Verification of computational science codes is dominated by testing. Testing remains the most essential contributor to the collection of verification evidence. Sufficient confidence in verification of software firmly rests upon the idea of sufficient testing. Inadequate testing increases the risk of malfunctioning software in important circumstances. Most really severe problems that arise in verification of computational science codes are not hard faults that create clear compiler failures, dramatic code crashes, or other clearly interpreted symptoms. Rather, these problems center on lack of accuracy of numerical solutions of partial differential equations that is directly (or indirectly) traceable to algorithmic failures. Algorithmic failures are mathematical problems, but also typically strongly correlated with the science in computational science practice. Detecting a lack of accuracy, as opposed to an outright code crash, can be very difficult (hence the apparent importance of "In the Eye of the Beholder" assessment that is too common). Extensive experience may be required to recognize a true algorithm failure. Lack of accuracy may also be due to lack of discretization resolution in a given calculation; the algorithm may be correct, but the number of finite elements, or finite difference resolution, or number of iterations specified for an iterative solver, may not be sufficient. Since HPC is dominated by the need to increase numerical resolution for hard problems where current achievable discretizations are known to be insufficient, we see how difficult it can be to detect true algorithmic faults or other code problems that don't result in hard software failures. Detecting an algorithmic failure may have to be deferred until numerical resolution issues can be dealt with. But notice that believed numerical resolution adequacy is itself coupled to some belief in current understanding of solution algorithms. To the extent that the algorithms may be wrong, the perception of needed numerical resolution may be wrong. Untangling these complex problems is not easy.

For these and other reasons, computational science testing ends up looking quite ad hoc over the very long run. In computational science, only a few of the existing test problems in the published literature in given subject matter areas are generally acknowledged to be

standards of simulation performance. If a standard test problem is identified (very rare) or implicitly exists (for example, the Sod problem in compressible fluid flow; Woodward and Colella, 1984), while the "correct solution" may be known there is no universal standard for how close to that answer a code must be for a given resolution to define success as opposed to failure. In other words, strong assessment principles remain absent even when good test problems exist and are applied.

Testing first and foremost depends upon having well-defined tests that a code passes or fails. While simple tests directed at individual code components can be devised that have strong assessment criteria, more complex tests that integrate larger parts of the physics and have greater numerical complexity are very difficult to devise, and can be extremely difficult to determine related assessment criteria. It is a critical problem in verification to devise such tests, as well as strong assessment criteria that create the verification consequences associated with the use of the test.

Benchmarks for code verification are needed for a wide range of physics and engineering application with special emphasis on coupled multi-physics. Important areas where solutions to semi-analytic verification test problems are sorely needed include, but are not limited to:

Component physics semi-analytic test problems and solutions in 1-, 2-, and 3-D. Examples include

- 2-D, 3-D hydrodynamics
- 2-D, 3-D radiation transport in specified medium
- 1-D multi-scale test problems for sub-grid scale and multi-scale problems
-

Coupled physics "semi-analytic" test problems and solutions in 1-, 2-, and 3-D. Examples of interest areas are:

- Coupled radiation-hydrodynamics
- Coupled neutron transport – hydrodynamics
- Coupled thermal-mechanical failures
- Coupled fire-thermal response phenomena
- Coupled large deformation mechanics and material "contact"

Semi-analytic test problems and solutions for radiation transport beyond flux-limited diffusion

- Angle dependent transport solutions

A desired goal is to achieve a state of sustainable code verification. This involves an infrastructure to maintain complex verification problems (on several meshes) under

configuration control and to rerun the entire verification test suite (with convergence studies) on demand and automatically report results to the desktop. A second goal would be the capability to report what features and capabilities were invoked in a specific application calculation and to compute metrics that measure the degree to which those features and capabilities are verified, both individually and all their interactions.

Research topics in the area of solution verification include:

- Practical methods for estimating or bounding numerical errors associated with spatial and/or temporal discretizations,
- Methods for estimating numerical errors associated with parameters that control the performance of numerical algorithms (e.g., artificial viscosity or hourglassing parameters), particularly in conjunction with other discretization errors, and
- Practical methods for making validation or application decisions with under-resolved models.

3.2 Validation

Well characterized validation experiments lie at the heart of simulation and model development. It is through these experiments that model accuracy can be assessed. Experiments can be generally classified into two types: (1) component (i.e. single physics phenomena); (2) integrated (i.e. coupled physics phenomena). The types of component and integrated validation experiments will vary from application to application. It is impossible here to give an all inclusive list of the subject areas and their corresponding validation experiments. Instead, the subject areas of interest are contained in the other white papers which interested PSAAP partners should consult. In addition, potential PSAAP institutions should keep in mind both existing and potentially new types of experiments which could serve code validation purposes. The sections below give representative examples of experiments that will be useful to the national laboratories. The purpose of the examples are to show the physics and engineering breadth and depth needed. It is not meant to be exhaustive.

3.2.1 Component experiments:

High quality experiments for component physics are needed for multi-scale, multi-physics, multi-dimensional codes. Due to the highly non-linear interactions that occur between physical processes in many applications, it is important to ensure that the isolated physical process under consideration be assessed for its accuracy. With integrated experiments, it is difficult to distinguish an error in the coupling between component physics from an error in the individual components themselves. This is called a compensating error. Therefore, component physics experiments form a critical part of any validation process.

A wide range of component physics experiments exist in the literature. Certain subsets of these are useful for validation purposes. ASC Alliance partners should leverage off of existing experiments where appropriate. A sample of component validation experiments is included below. As mentioned in the preface to this section, the list is by no means exhaustive.

- Numerical simulations of explosively driven deformations and high velocity impacts require validated models of material strength. The main challenge in constructing such models is the wide range of thermodynamic and mechanical conditions that occur in solid flow processes. Plate impact experiments with well diagnosed micro-structure are perfect examples of material strength validation experiments frequently used. These plate impact experiments are frequently gas-gun or laser driven.

- Granular flow is a multi-phase phenomenon of growing interest to the national laboratories. Simply put, granular materials are a large conglomerate of discrete macroscopic particles that may exhibit cohesive, electro-static or other types of forces. They exist in grain silos, powder metallurgy, planetary rings, etc. The academic literature abounds with experiments related to the behavior of granular materials under a variety of conditions. There are many validation experiments in this field due to their simple "table-top" like nature (Choi et. al, 2004).

- Hydrodynamic experiments can be broadly categorized into two classes: stable and unstable flow. Examples of stable hydrodynamics are laminar flow, explosively driven shocks and material deformation. Experiments performed in this regime seek to test hydrodynamic simulation capabilities. Unstable hydrodynamic flow involves the study of linear and non-linear instability growth to full scale turbulence in a variety of geometries.

- Equation of state measurements in both low pressure and high pressure regimes constitutes an important class of validation experiments. Gas-gun, laser, and Z-pinch facilities have all contributed to this field.

- Component radiation flow experiments are possible if the radiation flow is supersonic. That is, the material motion is decoupled from the propagation of the radiation front. In general there are two types of radiation flow experiments; local thermodynamic equilibrium (LTE) and non-LTE. In LTE, flow characteristics are represented by the Rosseland opacity. In non-LTE, besides a model of radiative transfer, atomic models are needed to obtain electron populations, which then describe absorption and emission processes. In LTE, there exist a variety of approximate descriptions of radiative transfer including flux-limited single and multi-group diffusion for which validation experiments in the diffusion regime are valuable. For non-LTE, where the full description of radiative transfer is needed, the validation of approximate models such as Eddington factors is valuable. Radiation flow experiments can and are performed at the Omega, Z and soon the NIF facilities (Perry, 1991).

- An important class of experiments deals with LTE flow of radiation through heterogeneous materials (Smith, 2003). The radiation flow in this type of medium is a theoretical challenge and models differ based on assumptions of character of the heterogeneity. Measuring effective mean free paths through mixtures would help validate theoretical models.

- Opacity is related to the transport of radiation and its coupling to matter and hence is closely related to the radiation flow experiments discussed previously. The accurate experimental determination of opacity is critical to validating the complex physics codes used to evaluate opacity. These codes rely on modeling atomic physics processes and rely on physics assumptions in order to make their computations feasible. Being able to validate the models feeding into the opacity codes ultimately leads to an assessment of the accuracy of the radiation transport model itself.

- Validating charged particle transport models is critical to understanding the slowing down of fusion reaction products in burning plasmas. Fortunately, there exist a wide variety of experiments looking at the attenuation and dispersion of ion beams in a weakly coupled plasma. One example is the energy loss of 1 MeV protons in a plasma target created by electric discharge in a hydrogen gas (Belyaev, 1996).

- Energy dissipation across a wide range of joints and other engineering interfaces.

Even with the large number of existing experimental data that helps validate the computational physics models, there is a real need for new experimental data in the following areas.

i. Experimental data on radiation flow in multi-dimensions in homogeneous media.

ii. Experimental data on planar and multi-dimensional flow radiation transport in heterogeneous media.

iii. Experimental data on fluid instabilities, material mixing and turbulent flows.

iv. Material fracture, failure and spall experiments.

v. Fluid instability experiments in non-planar geometries in systems with material strength.

vi. Charged particle transport and thermalization in strongly coupled plasmas.

vii. Thermonuclear reaction rates in strongly coupled plasmas.

viii. High pressure (Megabars to Gigabars) equation of state measurements.

ix. Dynamic failure of materials and engineered interfaces such as joints.

3.2.2 Integrated experiments

Ultimately, the applications under consideration tend to be multi-physics in nature. This means the validation of coupled/integrated physics models is of critical importance. In most codes, modularity of physics models means some type of operator splitting must be performed. Therefore, high quality well diagnosed experiments of integrated physics are needed for multi-scale, multi-physics, multi-dimensional codes. For example, theoretical models and simulations for radiative hydrodynamic systems abound, but they suffer from large uncertainties due to the complexity of the physics involved and the lack of experimental data.

The need for future experiments that validate integrated physics models is where the "rubber-meets-the-road". It is this class of experiments that ultimately any multi-physics code must be able to simulate. The list of experiments mentioned below is meant to be representative and not exhaustive. The purpose of the list is to convey a flavor of the integrated experiments needed by the national laboratories. Examples include but are not limited to:

i. Coupled radiation-hydrodynamics experiments including (1) radiative shocks and (2) photoevaporation front hydrodynamics.

ii. Radiation-hydrodynamic experiments in heterogeneous materials.

iii. Radiation-hydrodynamic experiments with turbulent flow.

iv. Thermal-mechanical failure of pressurized ensclsures.

v. Thermal-mechanical ignition and subsequent propagation in high explosives.

3.2.3 Virtual experiments and validation

An overlooked but interesting validation strategy is to use a simulation of the underlying micro-physics to help validate computational models. A simple example is direct numerical simulation of material strength properties via a molecular dynamics description of the fundamental processes (i.e. inter-atomic potentials). Frequently, physics models in codes are based on a number of physics approximations. In addition, it frequently occurs that the application of these models is to regimes that are inaccessible experimentally. The computational modeling of the fundamental physical processes and understanding how this micro-physical description translates into a continuum or macro description can be a powerful tool to help validate underlying assumptions in physics models.

3.2.4 Validation Methodology

Beyond interest in the validation of specific phenomena, there are a number of methological needs to support validation in a manner that allows quantification of uncertainties in non-linear, coupled multiphysics applications:

- Advanced statistical methods for making quantitative measurement/prediction comparisons, particularly in the presence of non-negligible variabilities and uncertainties in diagnostics, initial conditions, boundary conditions, and other model inputs.
- Tools to automate the process of quantitative validation.
- Methodologies for validation inference through a hierarchy of validation experiments ranging from simple material characterization test through a series of experiments of increasing complexity.
- Extrapolation inference from a validation parameter space to an application parameter space that is significantly outside the validation database
- Statistical methods for validation when there is only a single well instrumented test.

3.3 Uncertainty Quantification

Intelligent statistical sampling techniques will be necessary to sample the full domain of an N-dimensional space of possible outcomes. If the dimensionality is high (N>10) then standard sampling techniques (e.g. Monte Carlo) will not be nearly efficient enough to cover the full domain of uncertainty with a number of sample calculations (likely in 2-D) that are computationally feasible. Adaptive sampling procedures will have to be developed that will sample the full N-dimensional domain in an efficient enough way that clusters of sample simulations in those regions of the domain will capture where the sensitivity of the simulation response to variation in parameters, models, approximations etc. is highest. Examining the code response to the full variation all parameters in the physical models comprising the code by intelligent sampling of the N-dimensional parameter space will provide a total output certainty, but the full ensemble of models consisting of the combinations of the parameters and their variations may not satisfy data from available experiments. It thus becomes necessary to find the ensemble of models and the parametric settings that comprise them that at least satisfy available data. This requires an intelligent filtering of the full ensemble of models that cover all of the uncertainty space of the simulation. Once such a filtered set of parametric settings becomes available that give rise to an ensemble of output calculations that satisfy known experimental data from different experimental regimes, techniques must then be developed to propagate this set of models to regimes for which no experimental data exits and use the ensemble set in this regime to predict the total uncertainty of output metrics for those regimes.

The entire process of Uncertainty Quantification has important issues that must be addressed. The study of these issues in critical to any UQ component of a V&V program plan. Many of these issues are under current exploration in the laboratories V&V programs and strong collaboration of potential alliance centers with the laboratories programs in this area will be expected.

1. What approaches can be developed that allow for the determination of the dominant sensitivities in the code that drive the uncertainty in the output of a large scale simulation (particularly when the outputs are highly non-linear functions of the inputs).

2. How do these approaches compare with one another in determining the dominant sensitivity drivers of output uncertainty?

3. What approaches can be developed to propagate the uncertainty associated with a large number of uncertain parameters (N>>10) through the simulation to determine a prediction of the total uncertainty in the output metrics of large scale simulations. How can this be accomplished in a computationally efficient way when dimensionality of uncertainty space is high (i.e. N>>10) and the computation cost of a code run is very high?

4. How sensitive is the final uncertainty of output code metrics to the input probability density functions of the settings of code parameters in the physical models?

5. How many sample calculations are required to obtain the output uncertainty in a code simulation for an arbitrary number of uncertainty dimensions N? How can the accuracy of the output uncertainty for a given number of sample simulations covering an arbitrary number of uncertainty dimensions be quantitatively determined?

6. What techniques can be developed to determine if the ensemble of models that all fit known experimental data is complete?

7. What approaches can be developed to test a UQ methodology?

8. How do quantitatively determined output uncertainties compare when determined by different UQ methodologies?

9. What experiments can be designed that can be used to test a UQ methodology?

10. How can the confidence in a UQ methodology in the determination of the uncertainty in code output metrics be measured when possible experiments that could potentially test the methodology are not in the desired regime of code simulations?

11. Are there benchmark problems that can be developed that represent a fair test of competitive methodologies for UQ and sensitivity analysis?

12. What are practical methodologies for the aggregation and propagation of aleatory uncertainties, epistemic uncertainties, and combined alatory and epistemic uncertainties?

4. V&V Requirements

4.1 Summary of Important V&V Guidance

We summarize key conclusions from the above discussion:

C1: Validation calculations, validation experiments, and validation comparisons are subsets of potential sets of calculations, experiments, and experimental/computational comparisons that may be performed during a computational science project.

C2: Validation gaps are critically important to identify and communicate across the full scope of any computational science project.

C3: Careful documentation of needed validation experiments, whether they can or cannot be performed, is essential.

C4: As a logical structure, the validation plan points to the future (planned validation tasks) and is pointed to in the past (as validation tasks complete). The plan is a living entity and is probably sensitive to the accumulation of results from completed validation tasks.

C5: No experimental data = No validation.

C6: The first question that must be asked in any validation calculation that is compared with experimental data is: "Does the numerical error fatally corrupt the comparison with experimental data?" There are only three qualitative options for relevant answers to this question: (a) Yes; (b) No; (c) I don't know.

C7: The ***mathematical accuracy*** of validation calculations (***verification***) is not assessed by comparison with experimental data. Doing so is a logical fallacy.

C8: There is no validation comparison without reported experimental error bars. There are components of uncertainty in experimental error bars that are likely to be inadequately characterized, even though dedicated validation experiments should minimize this event. Reporting diagnostic fidelity of validation experimental data is a necessary condition so that a reported error bar can be constructed that has at least one sensible piece of information.

C9: Both computational uncertainty and experimental uncertainty are logically required in a validation comparison and should be reflected in the conclusions that are drawn. Providing a measure of computational uncertainty is a challenge.

C10: Don't do validation if you aren't willing to assess the consequences.

C11: Good enough for a journal does not imply good validation.

4.2 V&V Requirements

The V&V requirements follow under three general categories (1) Verification (2) Validation (3) Uncertainty Quantification. It is assumed that all PSAA codes have an underlying software quality check before the validation, verification, and uncertainty quantification studies are undertaken. This could mean regression testing for example.

Verification requirement:

Document and execute a plan for the development of verification test problems. The plan should address the definition and application of analytic and semi-analytic test problems and their solutions for both component physics and coupled physics in the code. The method of manufactured solutions should also be considered in building a verification plan. The plan should include a strategy for exchanging test problems and their results between alliance centers and ASC national laboratories.

Validation requirement:

Document and execute a validation plan, as discussed in Sections 2.3.1 and 2.3.2. The plan should detail what experimental data is presently available for validation, both of components and coupled physics in the code, for a specified application. Included in the plan should be a description of the approach to quantifying numerical accuracy for the validation calculations (see the sensitivity analysis and uncertainty quantification section below for details). That is, state how calculation error bars will be quantified and used in comparing simulation to experiment.

If the required experimental data do not exist, the plan should define the experiments that will be needed to validate components and couplings for the intended application. Included in the plan should be a description of how the new experiments will be performed, either by leveraging available ASC funds with existing in-house experimental facilities, or by placing awarded funds into existing partnerships (academic or with national labs) that have facilities sufficient to perform experiments that can acquire required experimental validation data. In addition, it is important that a process be set up by which the validation data produced be shared with the national laboratories..

Uncertainty quantification requirement::

Document and execute a sensitivity and uncertainty quantification plan, as discussed in Section 2.4 and 3.3. A description of how individual input factor uncertainties are rolled up into an overall output uncertainty should be included. Due to the difficult challenges facing uncertainty quantification, ASC PSAA centers are encouraged to explore new territories and approaches to the problem. This includes developing new software tools which can be shared with the national laboratories.

5. Conclusion

In a recent report (NSF, 2006) on the importance of computational science in engineering, the following finding was stated: "While verification and validation and uncertainty quantification have been subjects of concern for many years, their further development will have a profound impact on the reliability and utility of simulation methods in the future." (p. 36) The implication of this statement is that the promise of computational science can only be fulfilled by successfully performing verification and validation for critical applications. The NSF report emphasizes how difficult this challenge is.

These comments are equally important for ASC. This paper has been written to emphasize this statement. Computational science that is meaningfully applied carries the responsibility for its practitioners to engage in consequential V&V. From this perspective, our key conclusions in this paper can be briefly summarized.

1. As a community of computational scientists, we have the responsibility to perform experimental validation for important applications of computational science codes.

2. We have the responsibility to perform code and calculation verification. Validation is weakened by the failure to do so. This further threatens meaningful and credible application of our codes.

3. We have the responsibility to quantify both experimental and computational uncertainties if we seek to extract meaningful conclusions from validation activities.

4. We have the responsibility to document our plans, actions, and results when we perform V&V.

References

Note: The literature on Validation and Verification is vast. The references included below are a small subset of papers and books which cover various aspects of this field.

1. AIAA (1998), "Guide for the Verification and Validation of Computational Fluid Dynamics Simulations," AIAA Guide G-077-1998.
2. G. Belyaev (1996), "Measurement of the Coulomb energy loss by fast protons in a plasma target," Physical Review E, Volume 53, Number 3, 2701-2707.
3. J. Choi et. al (2004), "Diffusion and Mixing in Gravity-Driven Dense Granular Flows," Physical Review Letters, Volume 92, Number 17, 174301.

4. NNSA (2006), "Predictive Science Academic Alliance Program (PSAAP) – Guidelines for Applications of Interest."
5. National Science Foundation (2006), "Simulation-Based Engineering Science: Revolutionizing Engineering Science through Simulation," Report of the National Science Foundation Blue Ribbon Panel on Simulation-Based Engineering Science.
6. W. L. Oberkampf and T. G. Trucano (2002), "Verification and validation in computational fluid dynamics," Progress in Aerospace Sciences, Volume 38, 209–272.
7. W. L. Oberkampf, T. G. Trucano, and C. Hirsch (2004), "Verification, Validation, and Predictive Capability in Computational Engineering and Physics," Applied Mechanics Reviews, Volume 57, Number 5, 345-384.
8. T. Perry (1991), "Opacity measurements in a hot dense medium," Physical Review Letters, Volume 67, Number 27, 3784-3787.
9. M. Pilch, T. Trucano, J. Moya, G. Froehlich, A. Hodges, and D. Peercy (2000), "Guidelines for Sandia ASCI Verification and Validation Plans – Content and Format: Version 2.0," SAND2000-3101. (Available at http://www.sandia.gov)
10. S. G. Rabinovich (2005), Measurement Errors and Uncertainties – Theory and Practice, Springer.
11. P. J. Roache (1998), Verification and Validation in Computational Science and Engineering, Hermosa, Albuquerque.
12. A. Saltelli, K. Chan, and E.M. Scott (2000), Sensitivity Analysis, John Wiley & Sons.
13. C. C. Smith (2003), "Low Z opacities at high densities," Journal of Quantitative Spectroscopy & Radiative Transfer, Volume 81, 441-450.
14. T. Trucano, R. Easterling, K. Dowding, T. Paez, A. Urbina, V. Romero, B. Rutherford, and R. Hills (2001), "Description of the Sandia Validation Metrics Project," SAND2001-1339, August 2001. (Available at http://www.sandia.gov)
15. T. Trucano, M. Pilch, and W. Oberkampf (2002), "General Concepts for Experimental Validation of ASCI Code Applications," SAND2002-0341, April 2002. (Available at http://www.sandia.gov)
16. T. Trucano, M. Pilch, and W. Oberkampf (2003), "On the Role of Code Comparisons in Verification and Validation," Sandia National Laboratories, SAND2003-2752. (Available at http://www.sandia.gov)
17. United States Department Of Energy (2003), "Advanced Simulation and Computing Program Plan," Sandia National Laboratories report, SAND2003-3130P.
18. P. Woodward and P. Colella (1984), "The numerical simulation of two-dimensional fluid flow with strong shocks," Journal of Computational Physics, Volume 54, Number 1, 115-173.

Points of Contact

Richard Klein
Lawrence Livermore National Laboratory
klein4@llnl.gov

Scott Doebling
Los Alamos National Laboratory
doebling@lanl.gov

Frank Graziani
Lawrence Livermore National Laboratory
graziani1@llnl.gov

Martin Pilch
Sandia National Laboratory
mpilch@sandia.gov

Tim Trucano
Sandia National Laboratory
tgtruca@sandia.gov